# Executive (EXEC) Package
# Users' Guide

The MELCOR Executive (EXEC) package is responsible for overall execution control of MELGEN and MELCOR calculations. It coordinates various processing tasks for other MELCOR packages, including file handling, input and output processing, sensitivity coefficient modification, system timestep selection, time advancement, and calculation termination.

This document describes general input philosophy and the Executive package input required for MELGEN and MELCOR. Control directives for input processing are covered, and generic instructions for modification of sensitivity coefficients defined by other packages are given. Specific input requirements for phenomenological models in other packages are described in the users' guides for those packages. An introduction to the capabilities of MELCOR, its overall philosophy and design, and the MELCOR documentation set is given by the MELCOR Primer.

## Contents

## 1. Introduction

MELCOR executes in two parts. The first is a program called MELGEN, in which most of the input is specified, processed, and checked. When the input checks are satisfied, a restart file of all the information in the MELCOR database is written for the initial conditions of the calculation. The second part of MELCOR is the MELCOR program itself, which advances the problem through time based on the database generated by MELGEN and any additional MELCOR input.

MELGEN and MELCOR share a structured, modular architecture that facilitates the incorporation of additional or alternative phenomenological models. This structure consists of four primary levels. The top level is the Executive (EXEC) package, which consists of subroutines that coordinate the execution of various processing tasks and control the overall calculation sequence. The next level is composed of the database manager routines, which pass the required portions of the MELCOR database to the third level, containing the various modules or packages. Each package models a well-defined group of closely related phenomena or performs a well-defined special function. A unique two- or three-character alphanumeric identifier is assigned to each package and incorporated into the names of its subroutines and common blocks. The fourth level consists of routines performing utility-level functions such as matrix inversion, ODE integration, or transfer of mass and energy between packages. Many of these routines are contained in the MELCOR Utility (UTIL) package, which requires no user input.

In MELGEN, the Executive package calls the other MELCOR packages for each of several processing tasks that must be performed to set up and initialize the MELCOR database. The first task involves initializing all common block variables, including database pointers, sensitivity coefficients, and internal database variables, to default values. Next, all MELGEN input and output files are opened and the user input file is preprocessed to sort input records and store input data in a standard format ready for later retrieval by various packages. Input needed directly by the Executive package is then processed, and any sensitivity coefficient changes are made as prescribed by user input (see Sections 4, 5, and 7).

Most of the computational effort in MELGEN takes place during three passes of input processing. During the first pass, each package is called by the Executive package to read all the user-specified data for that package, to recalculate its database pointers for the size of the problem defined by the input, and to perform as much data initialization and error checking as is practical using only data within that package. During the second pass, each package is called by the Executive package to complete the initialization of its database with data defined in other packages during the first pass, and to perform consistency checks with other packages' data. The third pass initializes Control Functions (see the Control Function Package Users' Guide) that may depend on package data arguments defined during the second pass.

Input processing proceeds completely through each pass, but the next pass is not attempted if errors have been detected during a pass. After all input has been processed to the extent allowed, the Executive package calls each package to generate printed output and, if no fatal errors have been detected in any pass, to write the initial database into the restart file.

MELCOR execution involves two steps:

(1) a setup mode, during which the database (generated at time zero by MELGEN or at a later time by a previous MELCOR run) is read from the restart file and any additional input is processed, and

(2) a run mode, which advances the simulation through time, updating the time-dependent portions of the database each cycle and periodically writing the updated database into the restart file.

During setup mode, many of the tasks performed in MELGEN are repeated, including common block initialization, file opening, input preprocessing, sensitivity coefficient modification, and Executive package input processing. After the database is read from the restart file, any additional package input is read, processed, and checked with the database for consistency.

The heart of MELCOR is, of course, the problem advancement phase. The Executive package first checks to see whether it is time to end the calculation or to generate the next output edit, plot dump, or restart dump. If so, each active package is called in turn to perform the appropriate task. The Executive package then calculates the new system timestep (fixed for all packages), subject to the minimum and maximum specified by user input and according to requests by the packages themselves from the previous timestep.

Advancement across the Executive package timestep proceeds in three stages, with a loop through all active packages in each stage. The first stage allows all packages to initialize any interface of communication arrays needed by other packages during the advancement. The second stage performs the time advancement for each package, any of which may elect to subcycle across the Executive package timestep using smaller steps. A package may also elect to call for a fallback, in which case the Executive package repeats the entire current cycle with a smaller timestep. The third stage of the advancement allows each package to examine the results of calculations by other packages during the second stage. If calculated changes are excessive, a fallback may be called for at this point.

MELCOR's restart capability is extremely flexible. Each restart dump in the restart file contains all parts of the problem database required to restart MELCOR and continue the calculation. Restart dumps are generated by MELCOR at intervals specified by the user, based on either simulation time or central processing unit (CPU) time, and at the end of the calculation. Special restart dumps are also generated at key events such as vessel or containment failure. Sensitivity coefficients and selected MELCOR input variables can be changed at a restart to examine different calculation scenarios from that point, thus eliminating the need to repeat portions of the calculation that do not change. MELCOR restarts are exact; that is, a calculation restarted at some point with no change in input will continue along the identical calculational path that the code would have taken if it had not been stopped and restarted at that point.

One of the most important output files generated during a MELCOR run is the plot file, which contains data necessary to plot time histories of many of the variables calculated during the advancement. Although the volume of data involved is already substantial, there are many variables that a user *might* wish to plot that are not included in the file because their inclusion could make the file unmanageably large. Many of these variables are available as control function arguments, which are not directly plottable. In the past, users have been forced to deal with such variables by defining control functions "EQUAL" to the corresponding arguments. The values of these control functions, which are automatically included in the plot file, can then be plotted. (See the Control Function Package Users' Guide for details.)

Although this approach works, it is cumbersome, and often requires the definition of hundreds of control functions. Further, because all control functions must be defined in MELGEN, there is no way to add a variable to the plot file without rerunning the entire calculation. A capability has therefore been developed to allow the user to directly define a list of control function arguments to be added to the plot file in MELGEN, and to add to that list at the start, or at any restart, of MELCOR, under the control of the executive package. The necessary input is defined in Sections 4 and 5 of this Users' Guide, under the description of the PLOTxxx record.

## 2. MELGEN/MELCOR Input Format

Commands to MELGEN and MELCOR are contained in one or more input files. The file name is entered at the code prompt or (on most computer systems) as a parameter on the execution line.

A free-format processor (the INPPRE utility in the MELCOR Utility package) is used to preprocess all input. Every record must either have a valid identifier field, or must be an input directive record (which controls overall processing options), a comment record, or the terminator record.

The leftmost field is interpreted as an identifier field. After input processor directives have been processed, all active input records are sorted by identifier field so the input is order independent (as long as there are no duplicate identifier fields). Duplicate or "replacement" records are discussed below. The collocation sequence is:

0,1,2,...,8,9,A,B,C,...,X,Y,Z

Only alphanumeric characters may be used for identifier fields; upper and lower case letters are equivalent. User input data fields may be integers, logicals, single precision reals, double precision reals or character data. FORTRAN COMPLEX data is not supported. For example:

```
IDENT  0  1.  1.E28  1.D29  .FALSE.  ‘CHARACTER DATA’
```

Delimiters are blanks or commas. Multiple delimiters have the same effect as one delimiter. For example, the following three records are identical:

```
IDENT     4.4     5.5     6.6
IDENT,,,,,4.4,,,,,5.5,,,,,6.6
IDENT ,  4.4  ,  5.5  ,  6.6
```

Character data may be enclosed by single quotes. If there are no internal blanks or commas, and the string cannot be interpreted as a number, then the single quotes may be left off. For example, the following record contains an identifier field and four character data fields:

```
IDENT  BOY  ‘BIG BOY’  ‘1.1’  ‘MY,FAVORITE,YEAR’
```

All input is case insensitive. Lower case characters are automatically converted to upper case except for character strings enclosed by single quotes. For example, the following three statements are identical:

```
ident  george  1.e7
IDENT  GEORGE  1.E7
Ident  George  10000000.
```

When single quotes are used the situation changes. In the following the parameters are all different:

```
ident  george    filename
IDENT  ‘george’  ‘filename’
Ident  ‘George’  ‘Filename’
```

MELCOR packages access and interpret a specific input record by calling the CRACKR utility in the UTIL package and specifying all or part of the identifier field. CRACKR then locates the appropriate record and provides to the calling package the number of data fields and the type and value of each data field on that record. If the number of fields or their type (i.e., real, integer, logical, or character) does not agree with those expected, the calling package processes an error message.

A comment record has an asterisk (*) as the leftmost non-blank character. All characters to the right of an asterisk are interpreted as comments. Note, however, the command

*EOR* as described in Section 3 is not a comment. Completely blank lines are considered comments. An example of a comment line is:

```
* THIS RECORD IS JUST A LITTLE OLD COMMENT
```

A comment field may follow a non-comment field on the same record. For example:

```
IDENT  4.4  5.5   *   FORWARD AND REVERSE LOSS COEFFICIENTS
```

In some situations MELCOR uses multi-word parameters. In general the input processor will accept these quantities with blank separator in single quotes or - separator formats. For example, "STAINLESS STEEL OXIDE" can be referenced in a heat structure as

```
HS00011201  'STAINLESS STEEL OXIDE'
HS00011201   stainless-steel-oxide
hs00011201   Stainless-Steel-Oxide
```

In general, <u>lower case characters should not be used in multi-word parameters inside single quotes</u>, since they will then not be converted to upper case and may then not match the appropriate identifier or filename. See the individual package users' guides for details.

In some cases the identifier records allow optional trailing characters. This situation is denoted in this manual as

```
RESTARTF*ILE
```

For this input form, the code will recognize RESTARTF and RESTARTFILE as the same input. Only one form should be used in a given input set. This feature does not apply to the input processor directives in Section 3.

The last record in an input stream must be the terminator record. It has only a period and may optionally have a comment. For example:

```
.   * THAT'S ALL FOLKS
```

The default processing of duplicate records (those with the same identifier field) generates a fatal error. An input processor record "ALLOWREPLACE" permits a duplicate record to <u>replace</u> any and all records with the same identifier field that appears earlier in the input stream.

The format of each MELGEN or MELCOR input record is specified in the Users' Guide for the package to which that input records belongs. For example, the specification of records describing hydrodynamic control volumes will be found in the CVH Users' Guide, while that for records describing control functions will be found in the CF Users' Guide.

All such record specifications have the same general form. For example, a hypothetical input record might be specified as

**ABCnnnXX** - Object Definition Record

000 ≤ nnn ≤ 999 is the user number of some object
Optional

(1) NAME - User-defined name of the object
(type = character*20, default = none)

(2) VAR - Value of a variable names “VAR”
(type = real, default=0.0, units = kg/s)

This describes a record with an IDENT field ABCnnnXX in which the user-defined number of an object (a volume, flow path, or such) is embedded. The record may have one or two additional fields. The first is a CHARACTER name, of maximum length 20 characters, the second, which is optional, must contain a real number if it is present. If absent, the value 0.0 will be assumed. Thus, the record

```
ABC101XX   Peter 9.8
```

defines an object with the user number 101 and the associated name “Peter.” The variable VAR has been assigned a value 9.8. Similarly

```
ABC857XX   “Paul”
```

defines an object with the user number 857 and the associated name “Paul.” The default value of 0.0 will be assigned to the variable VAR. Note that none of the records

```
ABC123XX   ‘Extremely long, verbose name’   *NAME too long

ABC456XX   9.8                              *NAME wrong type

ABC789XX   Mary -1                          *VAR wrong type
```

is valid, for the reasons noted in the trailing comments. In addition, none of the records

```
ABC1Y2XX   “Tom” 37.

ABC12XX    Dick -4

ABC6789XX -4 Harry
```

will be recognized and processed as an ABCnnnXX record because the IDENT fields do not have the correct form.

## 3. Input Processor Directives

There are several input commands, termed input processor directives, that control the processing of the other input. For the most part, these commands appear as any other command except when an error occurs.

**ALLOWREPLACE** – Allow replacement records
Optional

Records with the same identification field are duplicates. The default is to treat these duplicate or replacement records as a fatal error. This command allows replacement records where only the last record in the input set with the duplicate field is retained. A warning message is generated by replacement.

The intended use of the replacement record feature is to permit changes in QA'd and other protected input sets without having to edit the input itself. Sensitivity studies and other changes can be completed without disturbing the base case input set. When this feature is desired, the ALLOWREPLACE command is required.

This record must be placed in the input set before any replacement records. It must also be in the active range defined by any *EOR* records as described below.

***EOR*** – Record to allow input to many codes in same file
Optional

Input sets are often moved between different computers and users in modern computer environments. All related input files for MELGEN, MELCOR, HISPLT, and other related programs can be combined into a single computer file by use of the *EOR* NAME directive. This command is designed to ensure that all the different input sets for a single problem do not get separated. The file structure is:

```
*EOR* MELGEN
   :
 MELGEN input with . terminator
   :
*EOR* MELCOR
   :
 MELCOR input with . terminator
   :
*EOR* HISPLT
   :
 HISPLT input
   :
```

```
*EOR* anything else
   :
 Ignored by MELGEN, MELCOR, and HISPLT
   :
{end of file}
```

The leading * must be in column 1 and the record is case insensitive. These commands are basically input on/off toggles. For example, MELCOR will process only data following an *EOR* MELCOR up to the next *EOR* record or end of file. Multiple *EOR* records with the same name are allowed. This permits deactivation of a section of input by bracketing it with *EOR* records. Any records before the first *EOR* will be processed (default processing mode is on). However, since the only records recognized by all three MELCOR processing codes (MELGEN, MELCOR, and HISPLTM) are comment records, only comments common to all three codes should normally appear before the first *EOR* record. Only records in the active range for the processing code being used are echoed to the code output listing. Other information outside of the active range is ignored.

If *EOR* records are not used in input, all records in the input file are processed.

Note that the *EOR* records are not comment records as discussed in Section 2 since they are recognized and processed by the preprocessor.

**R*I*F** – Redefine Input File

The R*I*F command allows the input set to be spread over multiple files. This record has the identifier field R*I*F followed by a filename. The file contains input data and the last record is a terminator record. The file is read into the input string at the current position. After the new file's terminator record is read, the new file is closed and reading continues with the next record in the original file. R*I*F directives may be nested eight layers deep. An input file might look like the following:

```
:
:
R*I*F    CVHDAT          * CVH INPUT DATA
R*I*F    CORDAT          * CORE INPUT DATA
R*I*F    RNDAT           * RADIONUCLIDE INPUT DATA
R*I*F    HSDAT           * HEAT SLAB INPUT DATA
:
```

## 4. MELGEN Input

During MELGEN execution, the Executive package either prompts the user for an input filename or reads the filename as an execution line parameter.

```
MELGEN fname*
```

It is noted that this might not work with an executable compiled with the Digital FORTRAN compiler. All the input for the MELGEN execution is in this file. If the user responds to the input prompt with a carriage return only (blank filename), then MELGEN attempts to read input from the file MELGIN. Only the TITLE record is required, but the DTTIME record is strongly recommended for use in MELGEN instead of MELCOR.

Like all input in the MELCOR suite of codes, each MELGEN input record has an identifier field followed by the input data fields, as discussed in Section 2. The MELGEN input records are listed in alphabetic order below.

In the following, the 80-character limit should be used cautiously because only the first 80 characters of a record are recognized. As a result, less than 80 characters actually are available because of the characters required for the variable name and any spaces.

**CRTOUT** – Edit format flag for 80-column output
Optional

Most package output routines will generate 80-column output if the CRTOUT record is present. This would be useful for examining output on an antique 80-column terminal in the unlikely event that anyone might still have such outdated article in his or her possession. If this record is not present, then some packages format output for 132-column line printers. There is no additional input for this record.

**DIAGF*ILE** – Filename for diagnostic output
Optional

The diagnostic output from MELGEN is written to the file named on this record. If this record is not present, the diagnostic output is written to MEGDIA.

(1) DIAGFILE - Filename for diagnostic output file.
(type = character*80, default = MEGDIA)

Note that the input processor by default converts lower case characters to upper case. If a lower case filename is desired, enclose the filename in single quotes.

**DTTIME** – Initial timestep
Optional

This record defines the initial timestep for MELCOR execution. If this record is not present, then an initial timestep of 1 s is used. MELCOR will use the timestep fallback capabilities if the initial timestep is not appropriate for the transient.

(1) DT - Initial timestep.
(type = real, default = 1.0, units = s)

**JOBID** – Job identifier
Optional

By default, each execution is assigned a unique computer-generated job identification. This record allows the user to override the default. The job identification appears in all output from the execution including plots from HISPLT.

(1) JOBID - Job identifier field.
(type = character*7, default = unique character string)

**OUTPUTF*ILE** – Filename for output listing file
Optional

The listing output from MELGEN is written to the file named on this record. If this record is not present, the output is written to MEGOUT.

(1) OUTPUTFILE
- Alternate filename for all listable output.
(type = character*80, default = MEGOUT)

Note that the input processor by default converts lower case characters to upper case. If a lower case filename is desired, enclose the filename in single quotes.

**PLOTxxx** – Control function argument to be plotted
xxx is any three-character alphanumeric string, used for sequencing only
Optional

These records may be used to add elements of the MELCOR database that are available as control function arguments to the plot file. Multiple records are required for arrays, because each record will add only a single variable. For example, plotting intact clad masses in multiple cells would require PLOTxxx records for the control function argument COR-MZR-CL.nnn for every cell (nnn) of interest. Variables added by PLOTxxx input records in MELGEN will be written to the plot file throughout the entire MELCOR calculation. Further variables may be added using PLOTxxx input records in MELCOR.

(1) CHARG - Database element identifier (name of control function argument), including any indices. Refer to the Users' Guides for the various packages for permitted values.
(type = character*24, default = none)

**RESTARTF*ILE** – Filename for restart file
Optional

The binary restart file for cycle zero is written to the file named on this record. If this record is not present, then the restart file is written to MELRST.

(1) RESTARTFILE
- Filename for cycle zero restart file.
(type = character*80, default = MELRST)

Note that the input processor by default converts lower case characters to upper case. If a lower case filename is desired, enclose the filename in single quotes.

**RUNONLY** – Run only on specified code version
Optional

In some user environments, MELCOR code versions can be changed without the knowledge of all users. This can create problems with parametric studies which require exactly the same code to produce meaningful results. This command requires that the code must be the same as that specified by a two-character base code version identifier (e.g., 'QK' for MELCOR 1.8.4), which is identified in header information written to terminal output and to the normal output file at the beginning of execution. If this condition is not true, the execution will terminate.

(1) ID - Two-character code version (e.g., 'NM').
(type = character*2, default = none, units = none)

**TITLE** – Title of the calculation
Required

Every calculation is required to have a title. This is written to the restart, edit, and plot files. The title on the MELCOR run must match a user-specified number of characters of the MELGEN title. If it contains blanks or if lower case characters are significant, the title must be enclosed in single quotes.

(1) TITLE - Title of the calculation.
(type = character*80)

**TSTART** – Initial start time
Optional

The problem time may be set to a nonzero initial value. All package input must be consistent with the problem time.

(1) TSTART - Initial problem time. Negative times are permitted.
(type = real, default = 0.0, units = s)

**UNDEF** – Redefine initialization of real database and scratch storage
Optional

This capability is of interest primarily to code developers, but the information is provided in the interest of completeness.

The real database for MELCOR is explicitly initialized in MELGEN to a value UNDEF. The same is done with the scratch storage area before each advancement by each package in MELCOR. By default, a value is used that is both recognizable and likely to cause a code abort; the intent is to increase the probability of detecting coding errors that result in undefined variables. On some systems, this value is the so-called IEEE "signaling Not a Number," and causes an interrupt when used as an operand in a floating point operation. On others, the value is -1.234E24. This input record allows the value of UNDEF to be changed, *but not set to 0.0*.

(1) IUND - INTEGER equivalent to the REAL value UNDEF
(type = integer, default = machine dependent, equivalence = UNDEF)

IUND=2143289344 ≡ UNDEF='7fc00000'X on some machines using IEEE
(default on HPW)
IUND=2140143616 ≡ UNDEF='7f900000'X on some machines using IEEE
(default on I60 and SUN)
IUND=32768 results in UNDEF='80000000'X on VAXVMS *only*
(default on VAXVMS)
IUND=0 results in UNDEF= -1.234E24 rather than 0.0
(default on other machines)

## 5. MELCOR Input

The MELCOR Executive package either prompts the user for an input filename or reads the filename as an execution line parameter.

MELCOR fname

It is noted that this might not work with an executable compiled with the Digital FORTRAN compiler. All the input for the MELCOR execution is in this file. If the user responds to the input prompt with a carriage return only (blank filename), then MELCOR attempts to read input from file MELIN. If the input file cannot be opened, then the user will be asked if he wishes to try again. Only the CPULEFT, CPULIM, TIMEk, and TITLE records are required, but many of the remaining optional records are recommended for user convenience and flexibility.

Like all input in the MELCOR suite of codes, each MELCOR input record has an identifier field followed by the input data fields.

**COMTC** – Number of matching characters in MELGEN and MELCOR titles
Optional

Each MELGEN and MELCOR execution must have a title. A user specified number of the characters on these titles must agree for the MELCOR calculation to proceed. If the titles do not agree to the specified number of characters, then the calculation is stopped. This prevents the user from accidentally running the wrong problem. It also allows him to change the MELCOR runs, use the same MELGEN restart file and change the end of his title to reflect the differences in the runs. COMTC must be greater than or equal to 20 and less than or equal to 80. If the input value is less than 20, then it is set to 20.

(1) NCOMTC - Number of characters to compare in the two titles.
(type = integer, default = 20, units = none)

**CPULEFT** – Desired minimum number of cpu sec left at end of calculation.
Required

The calculation will stop after completing a cycle if the number of CPU seconds left in the calculation is less than the number input on this record. This record is used to stop the calculation with enough time to save files, generate plots, etc. This applies mainly to systems where jobs are submitted to a batch queue with a time limit. The description of the input MELCOR variable CPULIM discusses the method MELCOR uses in determining the number of CPU seconds remaining in a calculation.

(1) CPULEF - Desired minimum number of CPU seconds left at the end of the calculation. CPULEF must be non-negative.
(type = real, default = none, units = s)

**CPULIM** – Maximum number of CPU seconds allowed for this execution
Required

The maximum number of CPU seconds allowed for this execution is defined by the value for CPULIM input on this record. However, for some environments, a job CPU time limit *external to MELCOR* may also be imposed (for example, the maximum amount of job time defined on a batch submission utility). For some computer systems, the Executive package is able to determine this limit using a system-dependent utility. When the CPU time used plus the safety factor CPULEF (input on record CPULEFT) is greater than or equal to the minimum of CPULIM and the externally imposed CPU limit, then a restart dump and edit will be written and the calculation terminated.

For interactive computer environments, CPULIM can be set to a large number and the calculation controlled by the interactive menu.

(1) CPULIM - Maximum number of CPU seconds allowed for this execution.
(type = real, default = none, units = s)

**CRTOUT** – Edit format flag for 80-column output
Optional

Most package output routines will generate 80-column output if the CRTOUT record is present. This would be useful for examining output on an antique 80-column terminal in the unlikely event that anyone might still have such outdated article in his or her possession. If this record is not present, then some packages format output for 132-column line printers. There is no additional input for this record.

**CYMESF** – Cycle Message Frequency Parameters
Optional

This record controls the frequency of the completed cycle messages written to the output and terminal files. One or two optional integer fields are allowed.

(1) NCYEDD - Number of cycles between messages written to the terminal or job stream file (unit 6). This can be reset from the interactive menu, as described in Section 9.
(type = integer, default = 10, units = none)

(2) NCYEDP - Number of cycles between messages written to OUTPUTFILE.
(type = integer, default = 1, units = none)

**DIAGF*ILE** – Filename for diagnostic output file
Optional

The diagnostic output from MELCOR is written to the file named on this record. If this record is not present, the diagnostic output is written onto MELDIA.

(1) DIAGFILE - Filename for diagnostic output.
(type = character*80, default = MELDIA)

Note that the input processor by default converts lower case characters to upper case. If a lower case filename is desired, enclose the filename in single quotes.

**DTINCR** – Timestep increase factor
Optional

MELCOR limits the increase in timestep between consecutive cycles to the ratio defined by this record. The default value allows an order of magnitude increase in timestep over five cycles.

(1) DTINCR - Maximum increase factor allowed in timestep.
Allowed range 1 < DTINCR < 2.
DT(next) ≤ DTINCR * DT(last)
(type = real, default = $10^{1/5}$ = 1.5848932, units = none)

**DTSUMMARY** – Additional timestep data to output file
Optional

This record extends the edit of $\Delta t$ information to the output file. One or two extra lines are produced for each cycle containing information on timestep requests by other packages. There is no additional input for this record.

**DTTIME** – Initial timestep
Optional

This record defines the initial timestep for this execution. If this record is not present, then the initial timestep read from the restart file is used.

(1) DT - Initial timestep.
(type = real, default = from restart, units = s)

Normally, DTTIME should be input only in MELGEN and not in MELCOR. No differences will result when starting a new calculation at cycle = 0, but when restarting with cycle > 0, DTTIME will reset the timestep from what would be obtained in a single calculation without the restart, and results will be different.

**EDITCF** – Special edit control function
Optional

This record specifies the number of a LOGICAL valued control function that will generate an edit if its value is .TRUE. This is intended to force an edit at the time of some user-specified event. (If the control function is .FALSE., it will not prevent generation of an edit if requested by some other mechanism.)

(1) IECF - This integer is the LOGICAL valued control function number.
(type = integer, default = none, units = none)

**EXACTTIMEk** – Define exact end-of-time-step times ($1 \leq k \leq 24$)
Optional

By default the timestep control does not attempt to "hit" any exact times in the determination of the values of $\Delta t$. The end-of-step time is simply the result of the summation of the $\Delta$ts to that time. This record tells the timestep control to reduce the timestep shortly before the specified time such that the end-of-step time matches the desired time exactly. The intended use for this record is to allow certain time-specified events, such as reactor scram, pump trip, or valve closure, to be matched exactly, thus eliminating a potential source of numerical sensitivity. Up to 25 times may be defined in any order.

(1) TIME - Desired time.
(type = real, default = none, units = s)

Warning: This calculation will not override the minimum timestep DTMIN defined on the TIMEk input record. If multiple exact times are defined too close together, the later exact time will be ignored.

**FORCEPLOT** – Control of extra plot information following timestep cuts
Optional

If there is a large timestep drop between consecutive cycles, MELCOR will generate plot dumps in addition to those defined by the TIMEk records described in this document. This feature is designed to capture the features of rapid events which occur with unknown timing. This record allows the user to change the trigger for this feature and the maximum number of dump sequences generated.

(1) DT_RATIO - Ratio of consecutive cycle timesteps to trigger extra plot dumps. If DT(i)/DT(i-1) < DT_RATIO, extra dumps are generated. DT_RATIO must be less than one.
(type = real, default = 0.2, units = none)

(2) MAX_NUMBER_OF_SEQUENCES
- Maximum number of extra plot dump sequences allowed per execution (i.e., the number of extra dump sequences generated is reset to zero when restarting a calculation).
(type = integer, default=20, units = none)

**JOBID** – Job identifier
Optional

By default, each execution is assigned a unique computer-generated job identification. This record allows the user to override the default. The job identification appears in all output from the execution including plots from HISPLT.

(1) JOBID - Job identifier field.
(type = character*7, default = unique character string)

**PLOTxxx** – Control function argument to be plotted
xxx is any three-character alphanumeric string, used for sequencing only
Optional

These records may be used to add elements of the MELCOR database that are available as control function arguments to the plot file on restarts. Variables named on PLOTxxx input records in MELGEN will continue to be written to the plot file. Up to 50 further variables may be added using PLOTxxx input records at a MELCOR restart, and will be included in the plot file for the duration of the execution in which they are read.

(1) CHARG - Database element identifier (name of control function argument), including any indices. Refer to the Users= Guides for the various packages for permitted values.
(type = character*24, default = none)

**MESSAGEF*ILE** – Filename for event message output
Optional
The event messages are written to the file named on this record. If the record is not present, then the messages are written onto file MELMES.

(1) MESSAGEFILE
- Filename for special messages.
(type = character*80, default = MELMES)

Note that the input processor by default converts lower case characters to upper case. If a lower case filename is desired, enclose the filename in single quotes.

**NOCOPY** – Suppress copy of files to output file
Optional

By default MELCOR will copy the message file (and diagnostic file in an error condition) to the end of the normal output file. This command suppresses the feature for those who prefer to keep these files separate.

**OUTPUTF*ILE** – Filename for output listing file
Optional

The listing output from MELCOR is written to the file named on this record. If this record is not present, the output is written onto file MELOUT.

(1) OUTPUTFILE
- Alternate filename for all listable output.
(type = character*80, default = MELOUT)

Note that the input processor by default converts lower case characters to upper case. If a lower case filename is desired, enclose the filename in single quotes.

**PLOTCF** – Special plot control function
Optional

This record specifies the number of a LOGICAL valued control function that will generate a plot dump if its value is .TRUE. If the control function is .FALSE., then a plot dump will still be generated if some other mechanism (e.g., time, physics package request) requests it.

(1) IPCF - This integer is the number of a LOGICAL valued control function.
(type = integer, default = none, units = none)

**PLOTF*ILE** – Filename for plot file
Optional

The binary plot data is written to the file named on this record. If this record is not present, the plot data is written onto the file MELPTF.

(1) PLOTFILE - Alternate filename for plot dump file.
(type = character*80, default = MELPTF)

Note that the input processor by default converts lower case characters to upper case. If a lower case filename is desired, enclose the filename in single quotes.

**RESTART** – Restart dump number used to start calculation
Optional

* * * NOTE: There are two forms of this command. * * *

Every dump on the restart file is associated with a cycle number and problem time. All information required to continue the calculation is contained in each dump. The dump generated by MELGEN is always cycle zero. The frequencies of dumps generated by MELCOR are specified on the TIMEk record series. A restart file dump is always generated when the calculation terminates gracefully, either at the time specified on the TEND record or when terminated by a MELCOR-detected error from which it cannot recover. On some systems, a restart dump will also be successfully generated on abnormal terminations (i.e., system-generated aborts or "bombs") through the use of system-dependent error recovery routines. The user may specify which dump in the restart file to use when starting the calculation. This allows the user to run part of the calculation, stop it, and then restart and continue the calculation. The user may also stop the calculation, change limited input parameters in order to run a variation in a sensitivity study, and continue the new calculation.

<u>FORM 1. - cycle number input</u>

(1) NREST - If $\geq 0$, Cycle number to start the calculation.
If -1 is input, the last restart dump is used.
If -2 is input, an interactive mode is entered.
The user will then be queried at each restart dump found on whether it is the desired restart point.
(type = integer, default = -1, units = none)

Example: RESTART -1 * restart from last cycle

<u>FORM 2. - calculational time input</u>

(1) TIME - the character string 'TIME'
(type = character, default = none)

(2) PROBLEM_TIME
- Calculation time to restart execution.
The code will restart from the first time on the restart file with time greater than or equal to the input.
(type = real, default = none, units = s)

Example: RESTART TIME 3600.0 * restart at first dump $\geq$ 3600. s

**RESTARTCF** – Special restart control function
Optional

This record specifies the number of a LOGICAL valued control function that will generate a restart dump if its value is .TRUE. If the control function is .FALSE., then a restart dump will still be generated if some other mechanism (e.g., time, physics package request) requests it.

(1) IRCF - This integer is the LOGICAL valued control function number.
(type = integer, default = none, units = none)

**RESTARTF*ILE** – Filename for restart file
Optional

The binary restart file to restart the calculation is specified by the file named on this record. If this record is not present then the restart file used is MELRST.

(1) RESTARTFILE
- Filename for restart file.
(type = character*80, default = MELRST)

Note that the input processor by default converts lower case characters to upper case. If a lower case filename is desired, enclose the filename in single quotes.

**RFMOD** – Restart file modification option
Optional

This record allows the user to copy selected restart dumps into a new restart file. User input history stored on the old restart file is also copied to the new file. This option allows the user to delete unnecessary restart dumps (i.e., dumps written at predetermined intervals for insurance but not actually needed to provide restart points) that merely consume file space. An arbitrary number of CYCLEN fields may follow the first field. No other input records are required if the RFMOD record is present, but the RESTARTFILE record may be used to specify the old restart file.

(1) NRESTF - File name for new restart file.
(type = character*80, default = none)

Note that the input processor by default converts lower case characters to upper case. If a lower case filename is desired, enclose in single quotes.

(N) CYCLEN - Cycle numbers of restart dumps to be written to new restart file. An arbitrary number of these may be included on this record. The last field may be "-1" to designate the final restart dump

without specifying its cycle number. An interactive option may be invoked by specifying "-2" for the first CYCLEN field instead of a list of cycle numbers. The user will then be queried for each restart dump found in the old restart file on whether it is to be included in the new restart file.
(type = integer, default = none, units = none)

**RUNONLY** – Run only on specified code version
Optional

In some user environments, MELCOR code versions can be changed without the knowledge of all users. This can create problems with parametric studies which require exactly the same code to produce meaningful results. This command requires that the code must be the same as that specified by a two-character base code version identifier, which is identified in header information written to terminal output and to the normal output file at the beginning of execution. If this condition is not true, the execution will terminate.

(1) ID - Two-character code version.
(type = character*2, default = none, units = none)

**SOFTDTMIN** – Define conditions for limited under-run of DTMIN input
Optional

There are models in some MELCOR packages which require a major reduction in timestep at some event. An example is the High Pressure Melt Ejection which may be activated at vessel failure. The exact timing of this event cannot be predicted by the user before running the calculation. The code requires a timestep at the start of ejection (e.g., $\Delta t \leq 10^{-4}$s) that is too small to use throughout the calculation. This input allows the user to accommodate this situation without restarting the execution and without specifying an excessively small DTMIN.

This input allows the timestep to be cut below the current DTMIN by a user-specified fraction for a limited number of times. Only cycles with $\Delta t$ less than DTMIN and where $\Delta t$ is decreasing from the previous cycle are counted. $\Delta t$ less than DTMIN but increasing is allowed as many times as necessary to get within the user-specified range. See the TIMEk input for user-defined limits.

(1) UNDERRUN_FRACTION
- This is the fraction of DTMIN which is allowed. $10^{-2}$ to $10^{-3}$ are normally acceptable. Acceptable values are from 1 to $10^{-8}$. The default condition does not allow under-runs.
(type = real, default = 1.0, units = none)

(2) NUMBER_OF_OCCURRENCES

- This is the number of occurrences allowed per execution. The counter is reset to zero at restart.
(type = integer, default = 0, units = none)

As an example, consider the situation where the code was running with DTMIN=0.01 and $\Delta t$ > DTMIN. The input record

SOFTDTMIN 0.005 3

would allow the timestep to be suddenly dropped to as low as 5 x $10^{-5}$s up to three times during the execution without terminating the run.

**STATUSF*ILE** - Filename for MELCOR status request file
Optional

To allow the user to get information from a "batch" execution, MELCOR will check for the existence of a 'status' file (using the FORTRAN INQUIRE statement). If the status file exists, MELCOR will then generate and release a file named "MELMAIL" in the directory where the code is executing with a short execution status summary. The status file is then deleted. The default name is "MELSTAT" (in the current directory). The file contents are unimportant since the file is never read. Also see STOPFILE to terminate batch execution.

(1) STATUSFILE - Filename for MELCOR 'status' file.
(type = character*80, default = MELSTAT)

Note that the input processor by default converts lower case characters to upper case. If a lower case filename is desired, enclose the filename in single quotes.

**STOPF*ILE** – Filename for MELCOR stop file
Optional

To allow the user to terminate a batch execution, MELCOR will check for the existence of a 'stop' file (using the FORTRAN INQUIRE statement). If the stop file exists, then an edit, a plot dump, and a restart dump are generated, the calculation terminates, and the stop file is deleted. The default name of the stop file is MELSTP. The file is never read, so the contents of the file are unimportant. Also see STATUSFILE for control of batch executions.

(1) STOPFILE - Filename for MELCOR 'stop' file.
(type = character*80, default = MELSTP)

Note that the input processor by default converts lower case characters to upper case. If a lower case filename is desired, enclose the filename in single quotes.

**TIMEk** – Timestep, edit, plot and restart control (1 ≤ k ≤ 24)
Required - at least one of these records is required

The maximum and minimum timesteps, edit, plot and restart frequency may be controlled using these records. The first input record determines when the remaining variables go into effect. Before that time the data on the previous record are used. After that time the data remain in effect until the problem time is greater than the time on the next record. The values on the last record remain in effect until the calculation finishes. MELCOR calculates its system timestep based on directives from the packages but it cannot take timesteps greater than the maximum timestep or smaller than the minimum timestep. The maximum timestep must be at least a factor of two larger than the minimum. The edit, plot and restart dump frequencies are controlled by these records. If a user specifies an edit frequency of 120 s, then every 120 s a new edit is generated. Any of the following can be changed from the interactive menu.

(1) TIME - Time the data on this record go into effect.
(type = real, default = none, units = s)

(2) DTMAX - Maximum timestep allowed during time interval.
(type = real, default = none, units = s)

(3) DTMIN - Minimum timestep allowed during time interval.
(type = real, default = none, units = s)

(4) DTEDIT - Edit frequency during this time interval.
(type = real, default = none, units = s)

(5) DTPLOT - Plot frequency during this time interval.
(type = real, default = none, units = s)

(6) DTREST - Restart frequency during this time interval.
(type = real, default = none, units = s)

(7) DCREST - Restart frequency based on CPU time during this time interval. This field is optional.
(type = real, default = $10^{10}$, units = s, minimum=100.0)

**TEND** – End of calculation time
Optional

The calculation will stop when the problem time is greater than or equal to TEND. The calculation may also terminate because the CPU time is exhausted, interactive termination is requested, or one or more packages request the calculation be stopped. A restart file is written at the last cycle before the calculation terminates. This variable can be reset from the interactive menu, as discussed in Section 9.

(1) TEND - End of calculation time.
(type = real, default = 5.4321 x $10^{20}$, units = s)

**TITLE** – Title of the calculation
Required

Every calculation is required to have a title. This is written to the restart, edit, and plot files. The title on the MELCOR run must match a user-specified number of characters of the MELGEN title (see the COMTC command). If it contains blanks or if lower case characters are significant, the title must be enclosed in single quotes.

(1) TITLE - Title of the calculation.
(type = character*80)

**UNDEF** - Redefine initialization of real database and scratch storage
Optional

This capability is of interest primarily to code developers, but the information is provided in the interest of completeness.

The real database for MELCOR is explicitly initialized in MELGEN to a value UNDEF. The same is done with the scratch storage area before each advancement by each package in MELCOR. By default, a value is used that is both recognizable and likely to cause a code abort; the intent is to increase the probability of detecting coding errors that result in undefined variables. On some systems, this value is the so-called IEEE "signaling Not a Number," and causes an interrupt when used as an operand in a floating point operation. On others, the value is -1.234E24. This input record allows the value of UNDEF to be changed, *but not set to 0.0*.

(1) IUND - INTEGER equivalent to the REAL value UNDEF
(type = integer, default = machine dependent, equiv =UNDEF)

IUND=2143289344 ≡ UNDEF='7fc00000'X on some machines using IEEE

(default on HPW)
IUND=2140143616 ≡ UNDEF='7f900000'X on some machines using IEEE
(default on I60 and SUN)
IUND=32768 results in UNDEF='80000000'X on VAXVMS only
(default on VAXVMS)
IUND=0 results in UNDEF= -1.234E24 rather than 0.0
(default on other machines)

**WARNINGL*EVEL** – Define level for warning and information messages
Optional

MELCOR can generate many forms of warning and information messages during an execution. These messages go to the output, terminal, and/or diagnostic files. In some cases, the volume of messages can be a problem. This record allows the user to control the volume and severity of the messages. Each message is assigned a level in the range 0 to 5. A fatal error is assigned the level 0. The least important messages (to the user in a successful execution) are assigned the level 5, and other messages are assigned intermediate levels according to relative importance. Messages with levels 4 and 5 are normally of interest only in the location of a code problem. This record tells the code to only record messages with a level equal to or less than the values input.

This feature applies only to the time advancement (run) mode of MELCOR. Messages generated during the setup mode (MELGEN) are always listed.

The range of each of the following is 0 to 5 except for terminal output. The terminal parameter allows an additional input of -1 to suppress messages from the message package being displayed on the terminal. This was the situation before MELCOR 1.8.2.

(1) MLEVO - Output file message level
(type = integer, default = 2, units = none)

(2) MLEVD - Diagnostic file message level. If input field is not present, then level is set to the output file level.
(type = integer, default = 2, units = none)

(3) MLEVT - Terminal message level. If input field is not present, then level is set to the output file level.
(type = integer, default = -1, units = none)

Note: Many of the MELCOR packages have not yet been modified to utilize this feature. The modifications will be included as rapidly as possible.

## 6. Plot Variables and Control Function Arguments

The elements of the Executive package’s database that may be used for plot variables and control function arguments are listed and described below. If the variable is a plot variable, then a ‘p’ will appear between the slashes ‘/’. If the variable is a control function argument, then a ‘c’ will appear between the slashes ‘/’.

TIME /pc/ Problem time.
(units = s)

DT /pc/ Timestep.
(units = s)

CYCLE /pc/ Cycle number.
(units = dimensionless)

CPU /pc/ CPU time used.
(units = s)

WARP /p/ Problem time minus initial problem time, divided by CPU time.
(units = dimensionless)

LOCALWARP /p/ Same as WARP except computed with decaying time history (i.e., calculated over a few cycles, weighted more heavily for the more recent cycles) to indicate current performance.
(units = dimensionless)

## 7. MELCOR Sensitivity Coefficient Modification

A goal for MELCOR is to implement as sensitivity coefficients all model or correlation constants and parameters, convergence criteria, and similar system parameters that are not normally accessible via input. These sensitivity coefficients are grouped into numbered arrays that represent a set of similar parameters, such as the several constants appearing in a single correlation. These arrays are identified by a four-digit number *nnnn* incorporated into the array names. Each MELCOR package is assigned a range of identifier numbers for the sensitivity coefficients in that package. The various sensitivity coefficients are described in the individual package users’ guides.

These sensitivity coefficients can be redefined in MELGEN or changed at any restart via MELCOR input (rather than modification of any code) so that the effects of modeling changes on the calculation can be examined. The sensitivity coefficient modification utility in the UTIL package coordinates the processing of any input that modifies the sensitivity coefficients. The value of any sensitivity coefficient is changed using an input record of the following form:

**SCiiiii** – Sensitivity Coefficient Identifier
00000 ≤ iiiii ≤ ZZZZZ is used for identifying and ordering the input

(1) NNNN - Unique four-digit identifier of the sensitivity coefficient array.

(2) VALUE - New value of the sensitivity coefficient. Values must be real-valued only - no integer values are allowed.

(3) NA - First index of the sensitivity coefficient

(4) NB - Second index of the sensitivity coefficient (if required)

The number of indices required will depend on the sensitivity coefficient.

For example, the following sensitivity coefficient input will change the default temperature of the control volume thermodynamics package from the value of 298 K to 368 K:

```
SC00000 2090 368. 1  *  CHANGE NATURAL TEMPERATURE TO 368 K
```

The Executive package does not have any sensitivity coefficients.

## 8. MELGEN and MELCOR Execution

MELGEN and MELCOR can be executed with or without command line parameters. The following forms are acceptable for MELGEN execution:

```
(1)    melgen
(2)    melgen  filename
(3)    melgen  i=filename
(4)    melgen  i=filename  id=runid
(5)    melgen  id=runid
```

It is noted that only (1) works with an executable compiled with the Digital FORTRAN compiler. In cases (1) and (5), the code will prompt for the input filename. The runid parameter is one or two alphanumeric characters which are appended to the named files created by the execution. The filenames are those discussed in relation to the commands DIAGFILE, OUTPUTFILE, RESTARTFILE, MESSAGEFILE, and PLOTFILE. No blanks are allowed around the "=" in the keyword=parameter format. The same form is used for MELCOR execution.

As an example, consider the MELCOR execution

```
melcor  i=myinput id=22
```

where the file myinput contains the following commands:

```
outputfile    'myout'
plotfile      'myplot'
restartfile   'myresf'
```

Execution generates the following files:

| | |
|---|---|
| myout22 | the output file |
| myplot22 | the plot file |
| myresf22 | the restart file |
| MELDIA22 | the default diagnostic file |
| MELMES22 | the default message file |

If filename extensions are used, such as

```
outputfile    'my.out'
```

the resulting filename would be

```
my22.out
```

Command line processing is always available on UNIX and PC-DOS systems. It will only function correctly on VAX-VMS if the MELGEN and MELCOR executables are installed as VMS DCL "foreign" commands.

## 9. MELCOR Interactive Interrupts

Interactive executions of MELCOR can be interrupted by certain messages typed at the terminal. Processed messages are:

STATUS! or ! for one-line status message
HELLO! for interactive menu
STOP!, QUIT!, END!, or EXIT! to terminate execution
OFF! to turn off interactive input (required for some debuggers to function correctly)

Commands are case-insensitive. Any character string not on this list will produce a brief list of recognized commands.

The interactive menu options are somewhat limited at the present time and will be expanded in future releases. Current options include:

(1) List DT data including statistical information

(2) Terminate calculation at end of cycle

(3) Interactive full edit (standard long edit with editor)

(4) Reset stop time

(5) Write full edit to separate file and release

(6) Reset terminal cycle message frequency

(7) Suspend execution for specified time

(8) Reset CPU limit

(9) Reset time interval input for timestep minimum and maximum limits, edit interval, plot interval, etc.

After the interactive options are completed, MELCOR returns to normal execution.

## 10. MELCOR Status from Batch Jobs

Many users prefer to run long executions as batch jobs to free their terminal for other functions. The ability to stop execution of these jobs is provided by the STOPFILE input in MELCOR. It is also possible to obtain the status of an executing batch job while it is running. The user can create a file specified by the STATUSFILE input record (default name "MELSTAT," with .DAT on VAX/VMS) in the directory where MELCOR is executing. The contents of the file do not matter; only the existence of the file is significant. MELCOR will detect and DELETE the MELSTAT file. MELCOR will then generate and release a file named "MELMAIL" in the same directory with a short execution status.

On most systems, only one such file is allowed at any time; a second mail request will overwrite the first message.

## 11. Example Input

The following example input illustrates the use of several of the MELGEN and MELCOR input records.

```
* Comments common to MELGEN and MELCOR
```

```
*
*EOR* MELGEN
TITLE         TMI-2
RUNONLY       NM
RESTARTFILE   tmirest
OUTPUTFILE    tmioutg
DIAGFILE      tmidiagg
DTTIME        0.1
CRTOUT
R*I*F         cvhinput
R*I*F         corinput
R*I*F         hsinput
R*I*F         rninput
R*I*F         otherinput
PLOT000       COR-MZR-CL.103
PLOTabc.......COR-MZR-CL.104
...
PLOTxyz       COR-MZR-CL.412
.
*
*EOR* MELCOR
TITLE         TMI-2
RUNONLY       NM
RESTARTFILE   tmirest
OUTPUTFILE    tmiout
DIAGFILE      tmidiag
MESSAGEFILE   tmimes
PLOTFILE      tmiplot
CPULEFT       30.0
CPULIM        36000.0
CRTOUT
NOCOPY
RESTART       0
TIME1            0.0    5.0   0.0001   500.0   2.0   500.0
TIME2         1000.0   10.0   0.0001  1000.0   5.0  1000.0
TEND          6000.0
R*I*F         scinput
PLOT000       RN1-ADEP-1-2-1.30001
PLOTabc.......RN1-ADEP-1-4-1.30001
...
PLOTxyz       RN1-ADEP-1-16-1.30001
.
```